Submitted to IEEE Transactions on Robotics and Automation for the special issue on Virtual Reality in Robotics and Automation.

Computer Vision Assisted Virtual Reality Calibration

Won S. Kim

Jet Propulsion Laboratory

California Institute of Technology

4800 Oak Grove Drive

Pasadena, CA 91109-8099

tel: (818)354-5047, fax: (818)393-5007

email: Won.S.Kim@jpl.nasa.gov

ABSTRACT

A computer vision assisted semi-automatic virtual reality (VR) calibration technology has been developed that can accurately match a virtual environment of graphically simulated three-dimensional (3-D) models to the video images of the real task environment. In conventional model-based computer vision, camera calibration and object localization are performed sequentially. This sequential update cannot compensate for the inaccuracy in initial camera calibrations. We have developed a new 20-variable weighted least-squares algorithm that updates both camera and object models simultaneously for given two camera views of two mating objects. This simultaneous update enables accurate model matching even with rough, approximate initial camera calibrations. The developed semi-automatic VR calibration supports automated intermediate updates, eliminating nearly all operator interaction except for initial coarse matching. In our quasi-static supervisory telerobotic applications, intermediate VR calibrations are performed intermittently at a few robot stopping poses only, as a cost-effective and safer approach compared to real-time visual servoing. Extensive experimental results comparing alignment errors under various viewing conditions are presented in the paper. Using the VR calibration technology developed, we have successfully demonstrated an orbital replacement unit (ORU) insertion task within

the required $\pm 1/4$ *in* and $\pm 3^\circ$ alignment precision.

Key Words: virtual reality, calibration, computer vision, model matching, simultaneous update, supervisory telerobotics.

1. INTRODUCTION

Graphic simulation of telerobotic operations using known 3-D geometric models of the remote-site task environment has been widely used for off-line task analysis and planning and also for operator training [15], [22]. However, use of graphic simulation during the on-line telerobotic operation has been limited due to the lack of accurate matching between the graphically simulated virtual environment and the real task environment. In the past decade or so, there have been considerable efforts to develop virtual reality (VR) calibration techniques that can accurately match simulated 3-D graphic models to video images of the real task environment to enable more reliable, high-precision telerobotic operations. An "operator-interactive" VR calibration that performs 3-D model matching using manually entered corresponding model and image points was developed earlier, and successfully demonstrated its potential usefulness with a remote servicing task between Jet Propulsion Laboratory (JPL) and NASA Goddard Space Flight Center (GSFC) in May 1993 [13], [14], [18]. The VR calibration was achieved by two sequential model matching procedures: interactive camera calibration followed by interactive object localization. After the calibration, the operator was provided with calibrated graphics overlay on actual video images for immediate human visual verification and monitoring. This "operator-interactive" VR calibration and video overlay technology has been transferred to industry, which is now commercially available [19]. Recently we have made substantial improvements by incorporating computer vision techniques, resulting in high-accuracy semi-automatic VR calibration. This new computer-vision assisted VR calibration supports automated intermediate updates, eliminating nearly all operator interaction except for initial coarse matching. Preliminary results were presented earlier [16], [17]. This paper describes this new semi-automatic VR calibration in greater detail including mathematical derivations and extensive new experimental results.

VR calibration enables supervisory control beyond manual teleoperation. So far, manual teleoperation [1], relying intensively on human vision and intelligence for direct joystick control of a remote

manipulator, has been the most viable solution to many non-repetitive remote manipulations dealing with man-made or natural objects. In supervisory telerobotic operation [26], the operator issues higher-level commands that can be executed autonomously whenever possible. Autonomous execution of higher-level commands generally demands model-based control. One main reason of a difficulty with model-based supervisory control is that the present computer vision technology has very limited capabilities in generating and updating 3-D models in accordance with the scene. By contrast, the human visual system has amazing capabilities in generating and updating internal 3-D cognitive models in the brain with abundant visual illusions for clear perception [28]. In an advanced supervisory telerobotic system architecture with a virtual reality interface, the system needs to update the virtual 3-D world model intermittently to support higher-level command executions. The computer-vision assisted VR calibration enables semi-automated accurate updates of the virtual 3-D world model.

An immediate potential application of the computer-vision assisted VR calibration is for International Space Station (ISS) robotics, since the camera viewing problem is a concern in ISS telerobotic operations and vision system assistance is needed for high-precision alignment. For instance, during the orbital replacement unit (ORU) insertion task, the end effector close-up camera view is occluded by the ORU, while the overhead and other cameras provide limited views. Due to this visual occlusion and limited viewing problem, it is often difficult to ensure baseline manual teleoperation to satisfy the alignment within the precision requirement reliably [25]. For example, the alignment requirement for ISS remote power controller module (RPCM) ORU (Fig. 1) insertion is $\pm 1/4$ *in* for each translation axis and $\pm 3^{\circ}$ for each rotational axis [3], [25]. The VR calibration presented in this paper enables high-precision alignment by utilizing known geometric object models and their salient straight line edges in matching 3-D graphic models to actual video images, not specifically requiring artificial vision targets or fiducial markings on object surfaces. Use of known geometric models permits the VR calibration to work well even with partially occluded and limited camera views. Use of natural geometric features of man-made objects such as object straight-line edges makes the VR calibration not only versatile but more robust under poor viewing and harsh lighting conditions, since vision targets attached on object surfaces are in general much more sensitive to camera viewing and lighting conditions compared to object-outline natural edges. Accurate positioning of vision targets is cumbersome and expensive. Some objects such as an RPCM receptacle simply do not have enough surface space to attach several

required vision targets on.

Another important advantage of VR calibration for ISS robotic applications is that its software does not have to be installed onboard. It can be installed on the ground as a cost-effective solution. With ground-based VR calibration, two control modes can be considered for ISS telerobotic operations: 1) ground-assisted onboard control and 2) ground remote control. In the ground-assisted mode, an on-board crew member performing such a task as ORU insertion is assisted by VR calibration on the ground. Video images received on the ground are used to perform 3-D graphic model matching through VR calibration. The relative position between the ORU and the receptacle is then sent to the on-board crew as a precision alignment aid. In the ground remote control mode, a ground operator controls the space manipulator system directly by issuing robot auto move commands, while an onboard crew member may monitor the robot motion. Supervisory control supported by VR calibration is essential for ground remote operation, since simple manual teleoperation has undesirable safety problems due to a typical 2-8 s round-trip communication time delay between a ground control station and the low Earth orbit.

In this paper, the concept of semi-automatic VR calibration with computer vision assists is introduced in Section 2. With a brief description of a local line detector in Section 3, a mathematical framework of line-based model matching is described in Section 4. Section 5 presents our key new development of the simultaneous update algorithm that calibrates both camera and object models simultaneously, enabling high-precision matching. Section 6 describes the operational procedure used for the high-precision ORU insertion task. Extensive experiments were performed to compare algorithms under various viewing conditions, and the results are described in Section 7. Section 8 is the conclusion.

2. SEMI-AUTOMATIC VIRTUAL REALITY CALIBRATION

Three-dimensional model-based computer vision for object recognition and localization has been studied extensively in the past several decades [2], [4], [6], [7], [9], [20], [24]. At present, 3-D model-based recognition that requires global searches is not yet robust enough to be fully automated for general robotic applications. On the other hand, 3-D model-based object localization for pose refinement requires only local searches from a given initial estimate, and can be automated with a sufficient reliability. For this reason, we have currently implemented semi-automatic VR calibration that consists of initial

operator-interactive coarse matching and subsequent automated fine matching procedures. Assuming no prior knowledge of camera calibration parameters and object poses, the initial coarse matching is based on the human operator's interactive inputs guided by superb human visual recognition. Two different operator interaction modes are provided for initial coarse matching: 1) point-click and 2) graphic model control, both using a mouse. In the point-click interface, the operator clicks on 3-D model points and their corresponding 2-D image points using a mouse. After the corresponding points entry, the system performs point-based camera calibration and object localization to complete the initial coarse matching. In the graphic model control interface, the operator uses a mouse to roughly align a graphic model to a video image. The point-click interface is generally simpler and faster for initial coarse matching with 6-degree-of-freedom (dof) position/orientation and camera focal length adjustments. The graphic model control interface is, however, often useful when only minor adjustments are needed. If initial approximate calibration estimates happen to be known, the operator-interactive initial coarse matching procedure can be skipped. After the initial coarse matching, automated fine matching is performed to achieve more accurate matching by using 3-D model-based computer vision algorithms. We use a newly developed simultaneous update computer vision algorithm that updates both camera and object models simultaneously based on a 20-variable least-squares method as described in this paper. This simultaneous update algorithm significantly increases the accuracy of 3-D model matching compared to the conventional object localization algorithm [9], [13], [20], [23] that does not compensate for inaccuracy in prior camera calibration.

The above semi-automatic VR calibration can be repeated at each intermediate robot stopping pose, for example, along the path to insertion, since the relative alignment precision of VR calibration increases further as the mating parts get closer to each other. In earlier operator-interactive manual VR calibration, it would be very time-consuming since the operator must enter the corresponding points all over again for each intermediate VR calibration. In the new computer-vision assisted VR calibration, intermediate VR calibrations can be done easily by automated fine matching with virtually no operator interaction, since all the camera and object models are already fairly accurate through the initial/previous calibrations. These intermediate updates are closely related to position-based visual servoing [11], [30], since they both perform repeated vision-based position updates. In our implementation, intermediate VR calibrations are performed intermittently at a few robot stopping poses only.

In telerobotic applications where the remote-site task environment is quasi-static (target objects are fixed during robot positioning), the intermittent update method has several advantages over real-time updates of visual servoing. It is cost-effective with low computational power and no special dedicated hardware required. It increases reliability and safety, since the operator has time to verify the alignment and correct it if necessary. It permits ground-based VR calibration in space telerobotic operations, since communication time delay is not a problem.

There are several reports on vision-based precise relative positioning using two or stereo camera views despite camera calibration errors [10], [27]. In any alignment tasks using two camera views, it is in general true that the alignment error decreases as the two mating parts gets closer to each other, regardless of the camera calibration errors. For instance, when the two image points are touching in two camera views, we can say that the two points are physically touching independent of camera calibration errors. However, if the objects are not supposed to touch at the desired alignment, accurate camera calibration will yield better alignment precision. In an ISS RPCM example, the receptacle entrance frame is $1/2$ *in* larger than the ORU in both the horizontal and vertical dimensions to allow $\pm 1/4$ *in* clearance. The simultaneous update algorithm presented in this paper tries to best match object models to two camera views by adjusting both camera and object models. When inaccurate camera calibration parameters are not corrected as in the conventional sequential update algorithm, alignment errors are larger as described in this paper.

3. LOCAL LINE DETECTOR

In computer vision, line features are easier and more reliable to detect compared to point features, and thus line-based model matching algorithms are employed in our implementation. In our current application, we assume that image lines to be detected are already adjacent to the projected model lines through human-operator-assisted initial coarse matching. We thus employed a local edge detector instead of a global one for computational efficiency and precision.

We currently use Gennery's weighted-average local line detector [8], which is an enhanced version of the nearest edge pixel detector [9]. For a given projected 2-D model line, the weighted-average line detector determines the weights and location of the nearby image line in a least-squares sense. The given model line is first divided into intervals of about 3 pixels each. For each interval, the line detector

searches edge elements at approximately right angles to the model line (the nearest multiple of 45° from the image axes) by applying Sobel edge operator along the search, and computes the weighted average of the edge elements found. This weighted-average computation for each interval results in one weighted-average edge point per interval. The line detector then performs a weighted least-squares fit of a line to these weighted-average edge points. First we define an $x'y'$-coordinate system such that the x'-axis is along the model line with its endpoints at (0 0) and (0 L). By denoting the location and weights of the weighted-average edge element for the j-th interval by $(x'_j\ y'_j)$ and w_j, respectively, we can determine the perpendicular distances from the two endpoints of the model line to the corresponding image line, h_1 and h_2, by the weighted least-squares method [9]. By noting that the image line can be described as

$$(1 - \frac{x'}{L})\ h_1 + (\frac{x'}{L})\ h_2\ =\ y', \tag{1}$$

the weighted least-squares solutions for h_1 and h_2 are determined by

$$\mathbf{w}_{img} = \sum_{j=1}^{K} \begin{bmatrix} 1 - \frac{x'_j}{L} \\ \frac{x'_j}{L} \end{bmatrix} w_j \begin{bmatrix} 1 - \frac{x'_j}{L} & \frac{x'_j}{L} \end{bmatrix}, \tag{2}$$

$$\mathbf{u} = \sum_{j=1}^{K} \begin{bmatrix} 1 - \frac{x'_j}{L} \\ \frac{x'_j}{L} \end{bmatrix} w_j y'_j, \tag{3}$$

$$\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \mathbf{w}_{img}^{-1}\mathbf{u}, \tag{4}$$

where $\mathbf{w}_{img}$ is a 2×2 weight matrix of the resulting image line measurements h_1 and h_2 values, and its inverse $\mathbf{w}_{img}^{-1}$ is the covariance matrix describing their variances and covariances. Typical values of $\mathbf{w}_{img}^{-1}$ indicate that standard deviations of the image line measurements usually range from 0.1 pixel for a good straight image line to 10 pixels to a poor one. In order to take into account the modeling errors, for example, due to inaccuracy in 3-D geometric models and nonlinearity in camera models, the systematic error component is added to the weight matrix computation.

$$\mathbf{w} = [\ \mathbf{w}_{img}^{-1}\ +\ \sigma_{sys}^2\mathbf{I}]^{-1}\ , \tag{5}$$

where $\mathbf{I}$ is the 2×2 identity matrix, and $\sigma_{sys} = 0.5$ pixel is chosen as a nominal value by observing average residual values of the line-based model matching least squares methods described in next Sections.

Once h_1 and h_2 are obtained, the endpoint locations of the detected image line in image coordinates can be computed. If $(u_1\ v_1)$ and $(u_2\ v_2)$ are the two endpoints of the projected 2-D model line, its length and cosine and sine values of the slope angle are

$$L = \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2}, \tag{6}$$

$$c = \frac{u_2 - u_1}{L}, \tag{7}$$

$$s = \frac{v_2 - v_1}{L}, \tag{8}$$

and the corresponding image line endpoints in image coordinates are

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} u_1 - h_1\ s \\ v_1 + h_1\ c \end{bmatrix}, \tag{9}$$

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} u_2 - h_2\ s \\ v_2 + h_2\ c \end{bmatrix}. \tag{10}$$

By computing the weights and endpoint locations of the image line given by (5), (9), and (10) for each visible model line, we can generate a list of corresponding model and image lines. Only visible model lines are used to avoid unwanted false matches. This list is then used in line-based camera calibration and object localization as described next.

4. LINE-BASED MODEL MATCHING

Kumar [20] showed that the least-squares algorithm for object localization that solves for the rotation and translation simultaneously [24], [29] yields much better parameter estimates in the presence of noisy data than another approach that solves for rotation first and then translation [21]. He further showed that the infinite model-line algorithm performs better than the infinite image-line algorithm when extracted image lines have significant broken segments. Our algorithm derived here corresponds to the infinite model-line approach in concept, but its mathematical derivations are generalized so that they can be used for both camera calibration and object localization that can handle multiple camera views. These unified derivations greatly help a simple, concise formulation of the simultaneous update algorithm of the next Section.

For a given 3-D object model point (x_m, y_m, z_m) in object model coordinates, its 2-D projection on the image plane (u, v) in camera image coordinates can be computed by

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \mathbf{M} \begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix}, \tag{11}$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \mathbf{V} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \tag{12}$$

$$u = -f\frac{x_c}{z_c}, \tag{13}$$

$$v = -f\frac{y_c}{z_c}, \tag{14}$$

where $\mathbf{M}$ transforms object model coordinates to world coordinates, $\mathbf{V}$ transforms world coordinates to camera viewing coordinates, and f is the camera focal length which is the distance from the lens center to the image plane. The camera focal length is equal to its lens focal length when the focus is at infinity. The 4×4 object pose transform $\mathbf{M}$ describes the object pose relative to the world reference frame. The inverse of the 4×4 camera viewing transform $\mathbf{V}$ describes the camera pose relative to the world reference frame. The above relations of (11) - (14) are often presented in a matrix form

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{PVM} \begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix} = \mathbf{CM} \begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix}, \tag{15}$$

where w is a homogeneous coordinate scale factor and $\mathbf{P}$ is a 3×4 perspective projection matrix

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}. \tag{16}$$

. In the above derivations, the camera imaging geometry is formulated by the pin-hole camera model, where the 3-D model point and its corresponding 2-D image point are related by linear perspective

projection without considering, for example, nonlinear distortion of lens optics. It is further assumed that the camera optical axis is perpendicular to the image plane, and passes through the center of the camera view with zero offsets of the image center from the optical axis. Square-pixel resolution is also assumed for the captured video images, having uniform scaling for both horizontal and vertical axes. The common 640×480 pixel resolution for the NTSC video image (4:3 aspect ratio) is a square-pixel resolution.

The relations described above for the perspective projection of a point can be directly applied to the line-based model matching. Since a 2-D projection of a 3-D model line is still a straight line, the projected 2-D model line can be simply computed by 2-D projections of the two endpoints of the 3-D model line. Let (u_1, v_1) and (v_1, v_2) denote the computed 2-D image plane projections of the two endpoints of a 3-D model line, (x_{m1}, y_{m1}, z_{m1}) and (x_{m2}, y_{m2}, z_{m2}), respectively.

$$w \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \mathbf{CM} \begin{bmatrix} x_{m1} \\ y_{m1} \\ z_{m1} \\ 1 \end{bmatrix}, \tag{17}$$

$$w \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = \mathbf{CM} \begin{bmatrix} x_{m2} \\ y_{m2} \\ z_{m2} \\ 1 \end{bmatrix}. \tag{18}$$

Further let (x_1, y_1) and (x_2, y_2) denote the two endpoints of the 2-D image line detected by the weighted-average local edge detector (Section 2). The normal distances from the image line endpoints to the projected 2-D model line are given by

$$h_1 = (Ax_1 + By_1 + C)/M, \tag{19}$$

$$h_2 = (Ax_2 + By_2 + C)/M, \tag{20}$$

where

$$A = v_2 - v_1 \tag{21}$$

$$B = u_1 - u_2, \tag{22}$$

$$C = u_2 v_1 - u_1 v_2, \tag{23}$$

$$M = \sqrt{A^2 + B^2}. \tag{24}$$

In the line-based model matching, the least-squares solution is obtained that minimizes the normal distances between the projected 2-D model lines and their corresponding actual 2-D image lines in the least-squares sense. The line-based model matching can be used for both camera calibration and object localization. In the camera calibration, we determine $\mathbf{C}$, or equivalently $\mathbf{V}$ and f, for given $\mathbf{M}$. If $\mathbf{M} = \mathbf{I}$ (identity matrix), the camera calibration is performed relative to the object model reference frame. Since the 4×4 camera viewing transform $\mathbf{V}$ can be equivalently represented by three translational displacements (x_C, y_C, z_C) and three rotational angles $(\alpha_C, \beta_C, \gamma_C)$, the unknown vector to be solved for the camera calibration is defined by 7 variables including the camera focal length:

$$\mathbf{x}_C = (x_C, y_C, z_C, \alpha_C, \beta_C, \gamma_C, f)^T. \tag{25}$$

Sometimes an accurate camera focal length is known, e.g., in a fixed focal-length camera, and the camera calibration in this case determines the six pose parameters only. In the object localization, we determine $\mathbf{M}$ for given $\mathbf{V}$ and f. Since the 4×4 object pose transform $\mathbf{M}$ can be equivalently represented by three translational displacements (x_M, y_M, z_M) and three rotational angles $(\alpha_M, \beta_M, \gamma_M)$, the unknown vector to be solved for the object localization is defined by 6 variables:

$$\mathbf{x}_M = (x_M, y_M, z_M, \alpha_M, \beta_M, \gamma_M)^T. \tag{26}$$

When M pairs of corresponding model and image lines are given, we have $2M$ equations

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} \mathbf{h}_1(\mathbf{x}) \\ \cdot \\ \cdot \\ \mathbf{h}_M(\mathbf{x}) \end{bmatrix} = \mathbf{0}, \tag{27}$$

where a 2×1 vector $\mathbf{h}_i(\mathbf{x})$ consists of two normal distances of (19) and (20) for the i-th corresponding model and image lines,

$$\mathbf{h}_i(\mathbf{x}) = \begin{bmatrix} h_{1i}(\mathbf{x}) \\ h_{2i}(\mathbf{x}) \end{bmatrix}. \tag{28}$$

Note that $\mathbf{x} = \mathbf{x}_C$ of (25) for camera calibration and $\mathbf{x} = \mathbf{x}_M$ of (26) for object localization. When $M > N/2$ where N is the number of variables of $\mathbf{x}$, the system is overdetermined and a weighted least-squares method can be applied to find $\mathbf{x}$ that minimizes the weighted sum of the squares of all $2M$ normal distances

$$f(\mathbf{x}) = \sum_{i=1}^{M} \mathbf{h}_i^T(\mathbf{x}) \mathbf{w}_i \mathbf{h}_i(\mathbf{x}), \tag{29}$$

where $\mathbf{w}_i$ is given by (5).

$$\mathbf{w}_i = \begin{bmatrix} w_{11i} & w_{12i} \\ w_{12i} & w_{22i} \end{bmatrix}. \tag{30}$$

By denoting

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{w}_M \end{bmatrix}, \tag{31}$$

(29) becomes

$$f(\mathbf{x}) = \mathbf{H}^T(\mathbf{x}) \mathbf{W} \mathbf{H}(\mathbf{x}). \tag{32}$$

Since $\mathbf{H}(\mathbf{x})$ contains nonlinear functions of $\mathbf{x}$, the nonlinear least-squares solution that minimizes $f(\mathbf{x})$ can be obtained by Newton-Gauss method, which is a combination of Newton's method and the least-squares method originated by Gauss. The k-th iteration can be described as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (\mathbf{J}^T(\mathbf{x}_k) \mathbf{W} \mathbf{J}(\mathbf{x}_k))^{-1} \mathbf{J}^T(\mathbf{x}_k) \mathbf{W} \mathbf{H}(\mathbf{x}_k), \tag{33}$$

where the Jacobian is defined as

$$\mathbf{J}(\mathbf{x}_k) = \frac{\partial \mathbf{H}(\mathbf{x}_k)}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{j}_1(\mathbf{x}_k) \\ \vdots \\ \mathbf{j}_M(\mathbf{x}_k) \end{bmatrix}, \tag{34}$$

$$\mathbf{j}_i(\mathbf{x}_k) = \frac{\partial \mathbf{h}_i(\mathbf{x}_k)}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{\partial h_{1i}(\mathbf{x}_k)}{\partial \mathbf{x}} \\ \dfrac{\partial h_{2i}(\mathbf{x}_k)}{\partial \mathbf{x}} \end{bmatrix}, \tag{35}$$

$$\frac{\partial h_{1i}}{\partial \mathbf{x}} = (x_{1i} \frac{\partial A_i}{\partial \mathbf{x}} + y_{1i} \frac{\partial B_i}{\partial \mathbf{x}} + \frac{\partial C_i}{\partial \mathbf{x}}) / M_i$$

$$-(A_i\frac{\partial A_i}{\partial \mathbf{x}}+B_i\frac{\partial B_i}{\partial \mathbf{x}})(h_{1i}/M_i^2), \tag{36}$$

$$\frac{\partial h_{2i}}{\partial \mathbf{x}}=(x_{2i}\frac{\partial A_i}{\partial \mathbf{x}}+y_{2i}\frac{\partial B_i}{\partial \mathbf{x}}+\frac{\partial C_i}{\partial \mathbf{x}})/M_i$$
$$-(A_i\frac{\partial A_i}{\partial \mathbf{x}}+B_i\frac{\partial B_i}{\partial \mathbf{x}})(h_{2i}/M_i^2). \tag{37}$$

A_i, B_i, and C_i are functions of u_{1i}, v_{1i}, u_{2i}, and v_{2i}, and their partial derivations can be found in [13]. By noting that $\mathbf{W}$ is a block diagonal matrix with each block being a 2×2 matrix, computation of (33) can be done more efficiently by using the relations

$$\mathbf{J}^T(\mathbf{x}_k)\mathbf{W}\mathbf{J}(\mathbf{x}_k)=\sum_{i=1}^{M}\mathbf{j}_i^T(\mathbf{x}_k)\mathbf{w}_i\mathbf{j}_i(\mathbf{x}_k), \tag{38}$$

$$\mathbf{J}^T(\mathbf{x}_k)\mathbf{W}\mathbf{H}(\mathbf{x}_k)=\sum_{i=1}^{M}\mathbf{j}_i^T(\mathbf{x}_k)\mathbf{w}_i\mathbf{h}_i(\mathbf{x}_k). \tag{39}$$

5. SIMULTANEOUS UPDATE OF CAMERA AND OBJECT MODELS

In conventional approach, camera calibration and object localization are performed sequentially. This sequential update assumes that the camera calibration provides sufficiently accurate camera calibration parameters for the subsequent object localization. Accurate camera calibration, however, generally requires a calibration fixture. Placing a calibration fixture whenever the camera parameters are changed, for example, due to camera pan, tilt, zoom, or focus control, is not practical for telerobotic applications. In our practical approach, an object with known geometric model that is naturally seen by the cameras during telerobotic operation is used for camera calibration. Let us consider a typical telerobotic task environment to perform parts mating of objects M_1 and M_2 using two cameras, C_1 and C_2. The camera calibration of C_1 can be performed by matching the object model M_1 to the camera view C_1. From (27),

$$\mathbf{H}_{C_1M_1}(\mathbf{x}_{C_1})=\mathbf{0}. \tag{40}$$

The least-squares solution $\mathbf{x}_{C_1}=(x_{C_1},y_{C_1},z_{C_1},\alpha_{C_1},\beta_{C_1},\gamma_{C_1},f_{C_1})^T$ can be computed for given object pose $\mathbf{M}_1$. Similarly, camera calibration of C_2 can be performed by matching the same object model M_1 to the other camera view C_2 to determine $\mathbf{x}_{C_2}=(x_{C_2},y_{C_2},z_{C_2},\alpha_{C_2},\beta_{C_2},\gamma_{C_2},f_{C_2})^T$.

$$\mathbf{H}_{C_2M_1}(\mathbf{x}_{C_2})=\mathbf{0}. \tag{41}$$

After the above two camera calibrations, the object localization of M_2 can be performed by matching object model M_2 to each camera view.

$$\mathbf{H}_{C_1M_2}(\mathbf{x}_{C_1}, \mathbf{x}_{M_2}) = \mathbf{0}, \tag{42}$$

$$\mathbf{H}_{C_2M_2}(\mathbf{x}_{C_2}, \mathbf{x}_{M_2}) = \mathbf{0}. \tag{43}$$

The least-squares solution $\mathbf{x}_{M_2} = (x_{M_2}, y_{M_2}, z_{M_2}, \alpha_{M_2}, \beta_{M_2}, \gamma_{M_2})^T$ can be computed for given $\mathbf{x}_{C_1}$ and $\mathbf{x}_{C_2}$. Note that $\mathbf{x}_{C_1}$ and $\mathbf{x}_{C_2}$ are intentionally added as function arguments in (42) and (43), since they are needed for the simultaneous algorithm.

Combining the above four equations with 20 unknown variables results in the simultaneous update algorithm for two objects with two views.

$$\mathbf{H}(\mathbf{x}) = \mathbf{0}, \tag{44}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{C_1} \\ \mathbf{x}_{C_2} \\ \mathbf{x}_{M_2} \end{bmatrix}, \tag{45}$$

where $\mathbf{x}$ consists of 7 variables of $\mathbf{x}_{C_1}$ for camera C_1, 7 variables of $\mathbf{x}_{C_2}$ for camera C_2, and 6 variables of $\mathbf{x}_{M_2}$ for object M_2. The object pose M_1 is fixed in this derivation, since one frame must be fixed to get a unique solution. With more than 10 corresponding model and image lines, the nonlinear least squares solution of (44) can be obtained by the Newton-Gauss method. Its Jacobian is given by

$$\mathbf{J} = \frac{\partial \mathbf{H}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{H}_{C_1M_1}}{\partial \mathbf{x}_{C_1}} & 0 & 0 \\ 0 & \frac{\partial \mathbf{H}_{C_2M_1}}{\partial \mathbf{x}_{C_2}} & 0 \\ \frac{\partial \mathbf{H}_{C_1M_2}}{\partial \mathbf{x}_{C_1}} & 0 & \frac{\partial \mathbf{H}_{C_1M_2}}{\partial \mathbf{x}_{M_2}} \\ 0 & \frac{\partial \mathbf{H}_{C_2M_2}}{\partial \mathbf{x}_{C_2}} & \frac{\partial \mathbf{H}_{C_2M_2}}{\partial \mathbf{x}_{M_2}} \end{bmatrix}. \tag{46}$$

We applied both sequential and simultaneous update algorithms to an RPCM-like ORU insertion task using two views (side and overhead views C_1 and C_2) for comparison. Both cameras were set by manual pan, tilt, zoom, and focus control. The camera focal lengths were approximately 50 mm (vertical field of view angle 5.5°) for the side camera and 25 mm (11°) for the overhead camera. The inter-camera angle between the two camera optical axes was approximately 50°. The side camera was

about 7.5 m away from the receptacle, and the overhead one was about 3.5 m away. First, the line-based camera calibration was performed for each camera by matching the ORU graphic model M_1 to its camera view. After the camera calibrations, the ORU graphic model was well aligned to its camera views. Thereafter the object localization was performed by matching the receptacle graphic model M_2 with its camera views. In Fig. 2, the receptacle localization was performed by using the side camera view only. The receptacle model pose was determined to best align the projected model and image lines in the least squares sense. Since only the side camera view (top window in Fig. 2) was used, the receptacle graphic model was not aligned in the overhead camera view (bottom). In Fig. 3, the receptacle localization was performed by using the overhead camera view (bottom) only, and this time the receptacle model was not aligned in the side camera view (top). Fig. 4 shows the video overlay after the receptacle localization was performed using both camera views. Note that the receptacle model was still visibly misaligned in both camera views. This poor alignment was mainly due to inaccuracy in camera calibrations. Note again that the ORU was used for camera calibration which provided less accurate and sparse input data, since an accurate camera calibration fixture was impractical to use. The inaccuracy in camera calibrations causes the subsequent object localization of the sequential update to be also inaccurate. Fig. 5 shows the model matching result obtained with the simultaneous update. Note that the receptacle model is very well aligned in both camera views. Unlike the sequential update, the simultaneous update algorithm updated both the camera and object models simultaneously, achieving accurate matching even with rough, approximate initial camera calibrations. This clearly demonstrates that the simultaneous update is essential to achieve high-precision model matching using two views.

6. OPERATIONAL PROCEDURE

The above line-based simultaneous update algorithm has been developed for automated fine matching to refine camera calibration and object pose estimates, assuming approximate initial camera calibration and object pose parameters are known. Note that the current algorithm employs a local edge detector that searches nearby image lines only, within approximately 20 pixels or so, for a given projected model line. Therefore, when initial camera and pose parameters are not known fairly accurately, an operator-interactive coarse matching procedure needs to be performed first before automated fine matching. In initial coarse matching, an operator clicks model points and their corresponding image

points by using a mouse. Once corresponding points are entered, point-based camera calibration followed by point-based object localization can be applied to determine the object pose as presented in our earlier paper [13]. This sequential update, however, does not compensate for initial camera calibration errors during the subsequent object localization, and initial camera calibrations need to be sufficiently accurate. In [13], the solution was to move the robot arm holding an ORU to three different poses to enter more corresponding points over a wider region. This calibration procedure is time-consuming and took about 10 to 20 minutes.

Our new solution is to use point-based simultaneous update algorithm, which is very similar to the line-based simultaneous update algorithm described in earlier Sections. It turns out that this new point-based simultaneous update no longer requires robot arm re-positioning to cover a wide region for data entry, since the algorithm compensates for initial camera calibration errors. Fig. 6 shows a display screen after an operator completes the corresponding points data entry for one view. The operator clicks a graphic model point on the top video overlay window by using a mouse, and then clicks the corresponding image point on the same window (graphic models disappear during the image point entry). During this data entry, the operator can adjust camera and graphic models using a mouse. The lower video image window shows all the corresponding image points entered so far. Five or more points for each of ORU and receptacle per camera view are usually desired. In Fig. 6, seven points for ORU and six points for receptacle are entered for this view. Note that corresponding points entered are well distributed on the video image. The operator data entry time for two camera views typically takes approximately 2 to 3 minutes in total. The current graphic model states set by the operator's mouse control are fed to the least-squares algorithms as initial states together with correspondence data. In the specific example of Fig. 6, all the least-squares algorithms of point-based camera calibration, object localization, and simultaneous update converged well to the desired solutions from the initial graphic model states shown in the picture. Even when the initial graphic model poses were intentionally set to more than 40 degrees off from the actual poses, they still all converged well. However, in some other camera viewing conditions with different initial poses, we noticed that the convergence range of the point-based 20-dimensional simultaneous update was often less wide than those of the sequential update. In order to maintain the convergence range as wide as the sequential update in general, point-based simultaneous update can be preceded by point-based sequential update of camera calibration and object localization for operator-

interactive coarse matching. Since computation times for least-squares methods are negligible compared to human data entry time, an inclusion of the optional sequential update does not reduce the operational efficiency. After the point-based simultaneous update, the automated lined-based simultaneous update algorithm can be applied for fine matching and pose refinement.

Here is the operational procedure used to demonstrate a high-precision ORU insertion into its receptacle. (1) Move the robot arm holding the ORU to an initial position. (2) Grab video images from two cameras showing both the ORU and the receptacle. (3) Enter corresponding points using a mouse for both views. (4) Apply point-based camera calibration and object localization sequentially. (5) Apply point-based simultaneous update. (6) Apply computer vision assisted line-based simultaneous update. This completes VR calibration at the initial ORU pose, and the system now knows the receptacle pose relative to the ORU. (7) Move the robot arm holding the ORU to the next via point towards the receptacle. (8) Grab video images from the same two cameras, and apply line-based simultaneous update (or object localization) for further pose refinement. No operator-interactive data entry is needed at this time, since fairly accurate estimates of camera/object variables are already known in previous VR calibration. As the ORU gets closer to the receptacle, new update increases the alignment precision. (9) Repeat the above intermittent update/pose refinement at next via points, until the ORU reaches at alignment ready for insertion. (10) Insert the ORU. This procedure was successfully used to demonstrate high-precision ORU insertion within the $\pm 1/4$ *in* and $\pm 3°$ alignment precision for various viewing and object pose conditions both at Jet Propulsion Laboratory and at NASA Johnson Space Center [12]. A video tape is also available [5].

7. EXPERIMENTAL RESULTS

A series of experiments were performed using an RPCM-like ORU insertion task. First, three sequential and one simultaneous update algorithms were compared by running each algorithm five times to measure the alignment errors. The ORU was 20 *in* away from its receptacle and the inter-camera angle between the two cameras used was approximately 90°. The alignment error was measured by comparing the true alignment pose with the algorithm estimate. The true alignment pose of the ORU relative to the receptacle was obtained by careful human visual alignment roughly within about 0.05 *in* and 0.5° precision. The position alignment error was computed as the vector sum of the translational

alignment errors for the three axes. Similarly, the orientation alignment error was computed as the vector sum of the rotational alignment errors for the three axes. The experimental results are shown in Fig. 7. The alignment error standard deviation from the true alignment was computed from five runs for each algorithm, and the computed standard deviations are connected by a dashed line. The position/orientation alignment error standard deviations were 1.57 in/2.69° for the sequential update using a single camera view C1, 1.64 in/4.12° for the sequential update using a single camera view C2, 0.51 in/1.19° for the sequential update using both camera views, and 0.21 in/0.88° for the simultaneous update using both camera views. In this experiment, the position alignment with the two-view simultaneous update was about 8 times more precise than that with the single-view simultaneous updates, and about 2.5 times more precise than that with the two-view sequential update. The orientation alignment with the simultaneous update was about 3-5 times more precise than that with the single-view sequential updates, and 1.4 times better than that with the two-view sequential update.

Second, the effect of the ORU-receptacle distance and the effect of the inter-camera angle on the alignment error were investigated together. The alignment errors were measured by running the simultaneous update algorithm five times each for four ORU positions of 20, 8, 4, and 2 in away from the receptacle at four different inter-camera angles of 90, 51, 28, and 14 degrees. The experimental results are plotted in Fig. 8 as a function of the ORU-receptacle distance. The same experimental results are plotted in Fig. 9 as a function of the inter-camera angle. Fig. 8 shows that the alignment error reduces as the ORU gets closer to the receptacle. This clearly illustrates that intermittent updates at intermediate robot stopping poses on a path to insertion increases the alignment precision. Fig. 9 shows that the alignment error reduces as the inter-camera angle increases towards 90° (orthogonal view). For each of the inter-camera angles at 28°, 51°, and 90° except for 14°, all five runs of the VR calibration satisfied the 1/4 in, 3° alignment precision requirement when the ORU reached at 2 in in front of the receptacle. The alignment precision was poor for the 14° inter-camera angle. The above experimental results indicate that the ORU insertion can be performed successfully within the 1/4 in, 3° alignment precision requirement when the inter-camera angle is greater than about 30 degrees.

Finally, the effect of the object image size (zoom) on the alignment error was investigated. In our setup, one camera happened to be equipped with a zoom lens, and the other with a fixed-focal-length lens. Three different viewing sizes of 10-, 20-, and 30-in-wide views were obtained by adjusting the

zoom of the zoom-lens camera and re-positioning the fixed-lens camera closer to or further away from the receptacle, so that each of the two camera views covers just enough to see both the ORU and the receptacle when they are 10, 20, and 30 *in* apart, respectively. The image sizes of the ORU and receptacle for the 10-*in*-wide view are larger than those for the 30-*in*-wide views. The inter-camera angle was set to 90°. As expected, the experimental results in Fig. 10 show that the alignment error increases as the object image size gets smaller.

Another way to investigate the alignment error is to compute 1-σ error ellipses from the covariance matrix $(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}$ resulting from the least squares solution. In order to have the resulting covariance matrix represent camera pose variances in camera frames while object pose variances in object frame, each iterative update of the least-squares method described in Section 4 and 5 needs to be performed in incremental form. In the incremental update, unknown variables $\mathbf{x}_C$ of (25) and $\mathbf{x}_M$ of (26) are associated with incremental adjustments of camera and object poses $\Delta\mathbf{V}$ and $\Delta\mathbf{M}$. In each iteration, $\mathbf{V}$ and $\mathbf{M}$ are updated by

$$\mathbf{V} \;=\; \Delta\mathbf{V}\;\mathbf{V}, \tag{47}$$

$$\mathbf{M} \;=\; \mathbf{M}\;\Delta\mathbf{M}, \tag{48}$$

and the initial conditions for the next iteration are set to $\Delta\mathbf{V} = \Delta\mathbf{M} = \mathbf{I}$ or $\mathbf{x}_C = \mathbf{x}_M = \mathbf{0}$. Note that $\Delta\mathbf{V}$ is pre-multiplied, while $\Delta\mathbf{M}$ is post-multiplied. If $\Delta\mathbf{V}$ is post-multipied and $\Delta\mathbf{M}$ is pre-multiplied, they both are expressed relative to the world reference frame (see (11) and (12)). From the covariance matrix $(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}$ resulting from the above incremental-form least squares solution, we can compute 1-σ error ellipses, which are 2-dimensional projections of the 20-dimensional error ellipsoid (18-dimensional if two camera focal lengths are given). For instance, 4 elements of the covariance matrix associated with variables x_i and x_j form

$$\Sigma_{ij} \;=\; \begin{bmatrix} \sigma_i^2 & \sigma_{ij} \\ \sigma_{ij} & \sigma_j^2 \end{bmatrix}, \tag{49}$$

and the 1-σ standard error ellipse is given by

$$(\frac{x}{\sigma_i})^2 \;-\; 2\rho(\frac{x}{\sigma_i})(\frac{y}{\sigma_j}) \;+\; (\frac{y}{\sigma_j})^2 \;=\; 1 \;-\; \rho^2, \tag{50}$$

where the correlation coefficient $\rho \;=\; \sigma_{ij} \;/\; \sigma_i\sigma_j$.

Error ellipses obtained are shown in Fig. 11 for the experimental condition shown in Fig. 5, where the ORU was approximately 10 *in* away from the receptacle. The standard deviations of the camera poses

along the camera axes (z axes) are very large for both cameras and their error ellipses for z axes (upper left and middle left plots of Fig. 11) are truncated. This is because the camera pose along the optical axis and the field of view angle are highly correlated, in particular when the camera lens has a small field of view angle (telephoto lens) with little perspective viewing effect. The standard deviation σ of the receptacle pose estimate (bottom plots of Fig. 11) is within 0.14 *in* along all three axes, indicating that 3σ for 95% confidence level is within 0.41 *in* in this experimental condition. As the ORU gets closer to the receptacle, the relative pose estimate gets more accurate. For example, when the ORU is less than 5 *in* away, the pose estimate was less than 1/4 *in* at 95% confidence level. These covariance error analysis results agree with the experimental results in Fig. 8. The covariance error analysis can provide a powerful tool in comparing pose estimate errors of different objects under various viewing conditions, without relying on extensive actual error measurement experiments. We could also use the non-weighted covariance matrix ($\mathbf{W} = \mathbf{I}$) for error analysis, by assuming uniform one-pixel standard deviation for the image edge measurements. In our various experimental conditions, this assumption appears to be fairly reasonable. As an example, for the experimental condition shown in Fig. 5, the root-mean-squared residual of the simultaneous update least squares method was 1.2 pixels, and the maximum residual was 3.7 pixels.

8. CONCLUSION

We presented an exciting new technology of computer vision assisted semi-automatic VR calibration for reliable, high-precision telerobotic servicing. In particular, the newly developed simultaneous update of both camera and object models were described in great detail as a key new technique to produce high precision alignment. Experimental results indicate that the simultaneous update yields considerably more precise matching than the conventional sequential update that does not compensate for inaccurate camera calibration parameters. Experimental results also indicate that intermittent updates at a few intermediate robot stopping poses increases the alignment precision further. This semi-automatic VR calibration provides a new way of performing more reliable and accurate telerobotic servicing with model-based supervised autonomy beyond simple manual teleoperation.

ACKNOWLEDGMENT

This work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The author would like to thank Dr. Donald Gennery for providing weighted-average line detector and covariance analysis, Eugene Chalfant for conducting experiments, and Edward Barlow for building RPCM-like ORU mockup at Jet Propulsion Laboratory. The author also would like to thank Lucien Junkin and Ivan Spain at NASA Johnson Space Center (JSC) for their collaboration in experimental evaluation. Some of the video images used in this paper were taken from the JSC Automated Robotic Maintenance for Space Station (AMRSS) facility. The reviewers' comments were very helpful.

References

[1] Bejczy, A. K. (1980). Sensors, controls, and man-machine interface for advanced teleoperation. *Science*, 208(4450), 1327-1335.

[2] B. Bhanu, "CAD-based robot vision," Computer, vol. 20, no. 8, pp.13-16, 1987.

[3] D. Brown, M. Hiltz, A. Samji, and C. Thorton, MSS On-Orbit Operations Concept Document, SPAR-SS-R-044, vol. II, Appendix A. "RPCM ORU Exchange with SPDM," Dec. 1992.

[4] R. T. Chin and C. R. Dyer, "Model-based recognition in robot vision," ACM Computing Surveys, vol. 18, no. 1, pp. 67-108, 1986.

[5] Conference Video Proceedings, Int. Conf. Robotics and Automation, 1998.

[6] O. D. Faugeras and M. Hebert, "The representation, recognition, and locating of 3-D objects," Int. J. Robotics Research, vol. 5, no. 3, pp. 27-51, 1986.

[7] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: a paradigm for model fitting with applications to image analysis and automated cartography," Comm. ACM, vol. 24, no. 6, pp. 381-395, 1981.

[8] D. B. Gennery, "Improved edge measurement for visual tracking," Jet Propulsion Laboratory Internal Document, to appear.

[9] D. B. Gennery, "Visual tracking of known three-dimensional objects," Int. J. Computer Vision, vol. 7, no. 3, pp. 243-270, 1992.

[10] G. D. Hager, "A modular system for robust positioning using feedback from stereo vision," IEEE Trans. on Robotics and Automation, vo. 13, no. 4, pp. 582-595, 1997.

[11] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," IEEE Trans. on Robotics and Automation, vo. 12, no. 5, pp. 651-670, 1997.

[12] L. Junkin, "International Space Station robotic technology transfer program: Evaluation of calibrated synthetic viewing technology to augment ISS camera views for ORU insertion tasks," Phase II, AR&SD-98-001, NASA Johnson Space Center, 1998.

[13] W. S. Kim, "Virtual Reality Calibration and Preview/Predictive Displays for Telerobotics," Presence, MIT Press, vol. 5, no. 2, pp. 173-190, 1996.

[14] W. S. Kim and A. K. Bejczy, "Demonstration of a High-Fidelity Predictive/Preview Display Technique for Telerobotic Servicing in Space," IEEE Trans. on Robotics and Automation, vol. 9, no. 5, pp. 698-702, 1993.

[15] W. S. Kim and A. K. Bejczy, "Graphics displays for operator aid in telemanipulation", IEEE Int. Conf. on Systems, Man, and Cybernetics, pp. 1059-1067, Charlottesville, VA, pp. 1059-1067, 1991.

[16] W. S. Kim, D. B. Gennery, and E. C. Chalfant, "Computer Vision Assisted Virtual Reality Calibration," IEEE Int. Conf. on Robotics and Automation, pp. 1335-1340, Albuquerque, NM, Apr. 1997.

[17] W. S. Kim, D. B. Gennery, and E. C. Chalfant, L. Q. Junkin, I. M. Spain, S. B. Rogers, "Calibrated Synthetic Viewing," American Nuclear Society (ANS) Seventh Topical Meeting on Robotics and Remote Systems, pp. 596-602, Augusta, GA, Apr. 1997.

[18] W. S. Kim, P. S. Schenker, A. K. Bejczy, S. Leake, and S. Ollendorf, "An Advanced Operator Interface Design with Preview/Predictive Displays for Ground-Controlled Space Telerobotic Servicing," SPIE Conference 2057: Telemanipulator Technology and Space Telerobotics, pp. 96-107, Boston, MA, Sep. 1993.

[19] W. S. Kim, H. Seraji, P. Fiorini, R. Brown, B. Christensen, C. Beale, J. Karen, and P. Eismann, "Commercialization of JPL Virtual Reality Calibration and Redundant Manipulator Control Technologies," Third Int. Symp. on Artificial Intelligence, Robotics, and Automation for Space, pp. 23-26, 1994.

[20] R. Kumar and A. R. Hanson, "Robust Methods for Estimating Pose and a Sensitivity Analysis," CVGIP: Image Processing, vol. 60, no. 3, pp. 313-342, 1994.

[21] Y. Liu, T. S. Huang, and O. D. Faugeras, "Determination of camera location from 2D to 3D line and point correspondences," IEEE Conf. on Computer Vision and Pattern Recognition, pp. 82-88, 1988.

[22] P. B. Loftin and P. J. Kenny, "Training the Hubble space telescope flight team," IEEE Computer Graphics and Applications, vol. 15, no. 5, pp. 31-37, 1995.

[23] D. G. Lowe, "Fitting parameterized three-dimensional models to images," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 13, no. 5, pp. 441-450, 1991.

[24] D. G. Lowe, Perceptual Organization and Visual Recognition, Kulwer Academic Publishers, 1985.

[25] M. Muchinsky, E. Mohr, and B. Cura, "Remote power controller module (RPCM) orbital replacement unit (ORU) hardware to robotic systems integration standards (RSIS) verification test report," Oceaneering Space Systems 061-ER-GF1, 1996.

[26] Sheridan, T. B. (1192). *Telerobotics, Automation, and Human Supervisory Control*, Cambridge, MA: MIT Press.

[27] S. B. Skaar, W. H. Brockman, and W. S. Jang, "Three-dimensional camera space manipulation," Int. J. Robotics Research, vol. 9, no. 4, pp. 22-39, 1990.

[28] L. W. Stark, "How virtual reality works: the illusions of vision in real and virtual environment," SPIE Symposium on Electronic Imaging: Science and Technology, San Jose, CA, Feb. 1995.

[29] T. Tanabe, E. Ohyama, and H. Koyama, "Model based vision system for autonomous teleoperated spacecraft," American Astronautical Society/Japanese Rocket Society Joint Int. Symp., AAS 85-661, 1985.

[30] W. J. Wilson, C. C. Williams Hulls, G. S. Bell, "Relative end-effector control using Cartesian position based visual servoing," IEEE Trans. on Robotics and Automation, vo. 12, no. 5, pp. 684-696, 1997.

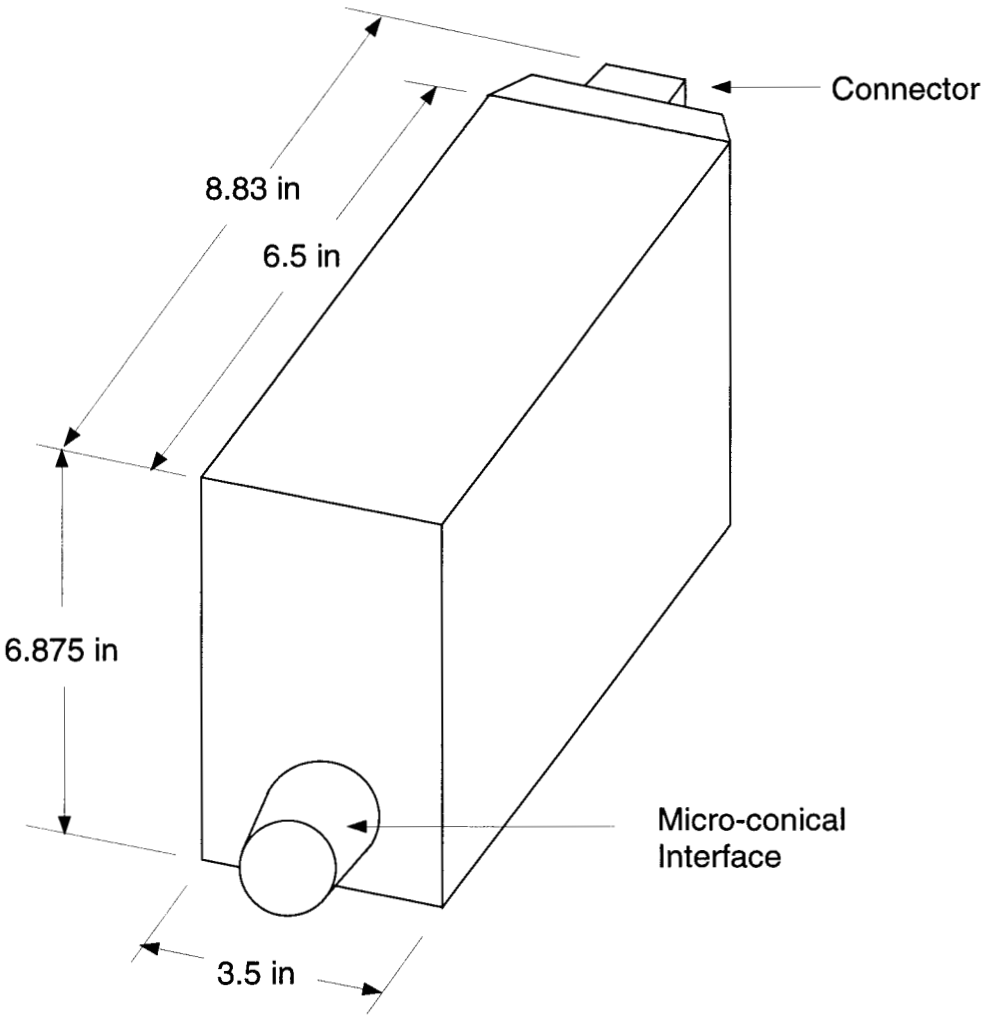

Figure 1: A schematic of the remote power controller module (RPCM).

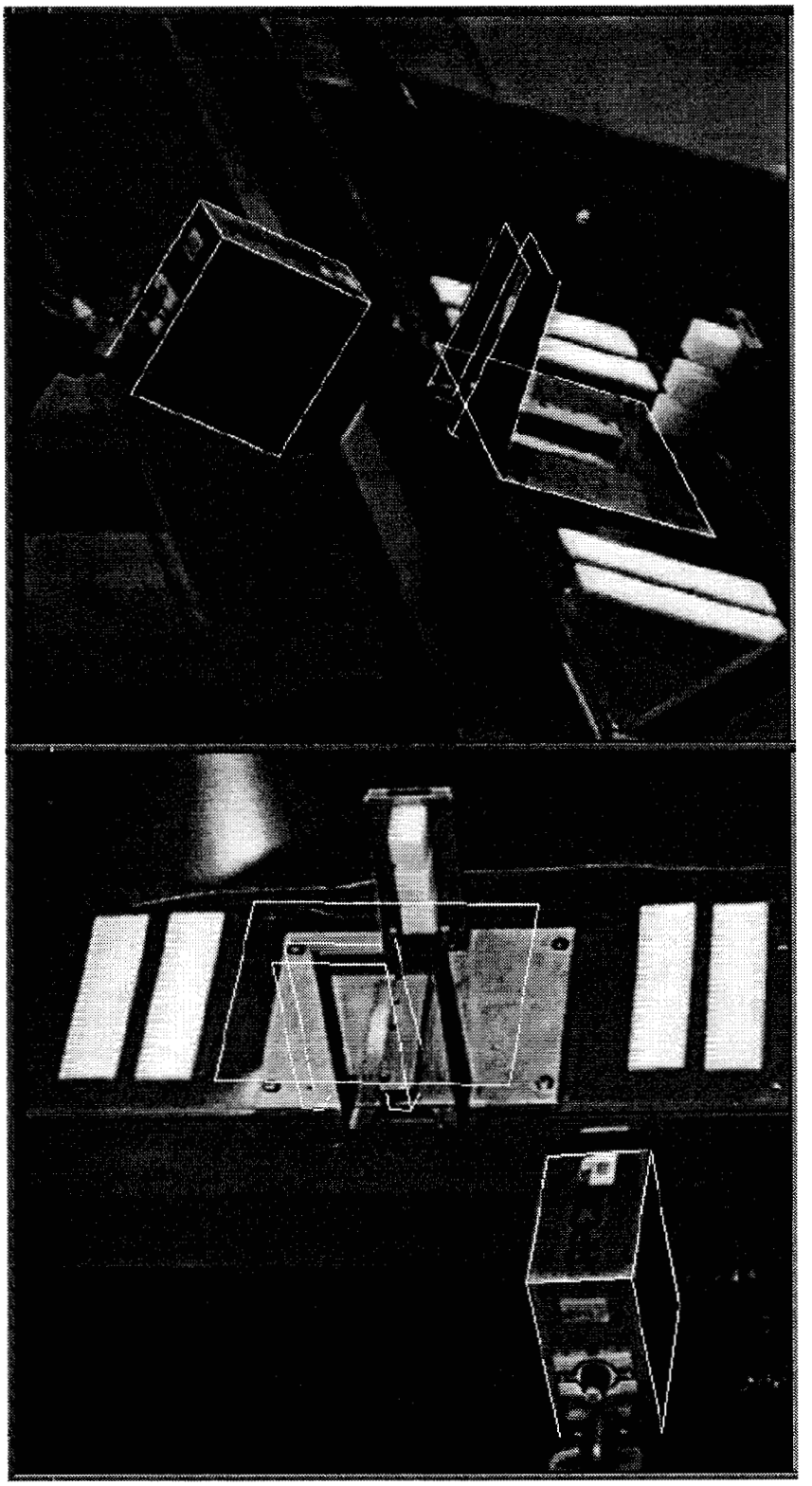

Figure 2: Receptacle localization using the side camera view (top) only.

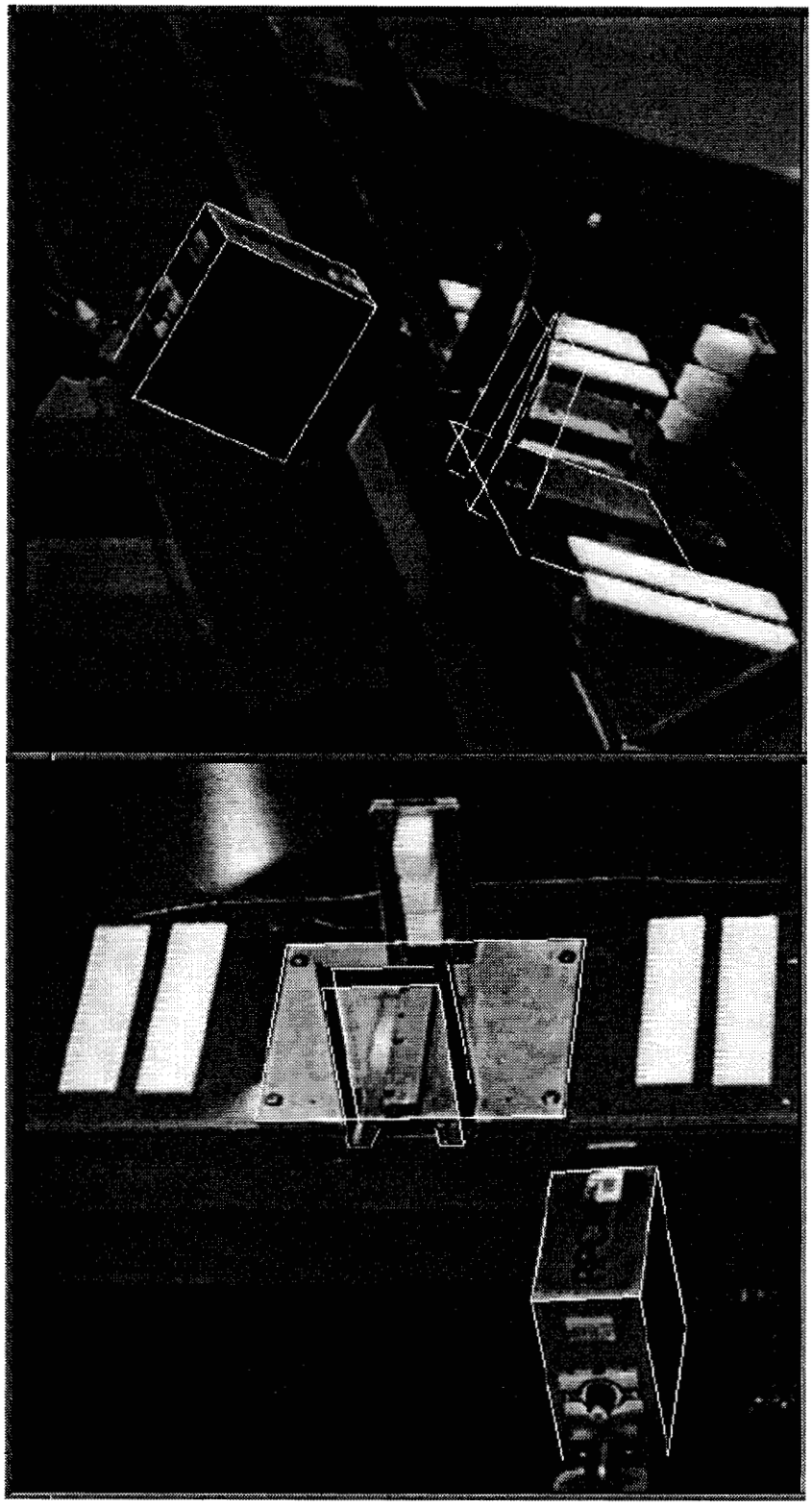

Figure 3: Receptacle localization using the overhead camera view (bottom) only.

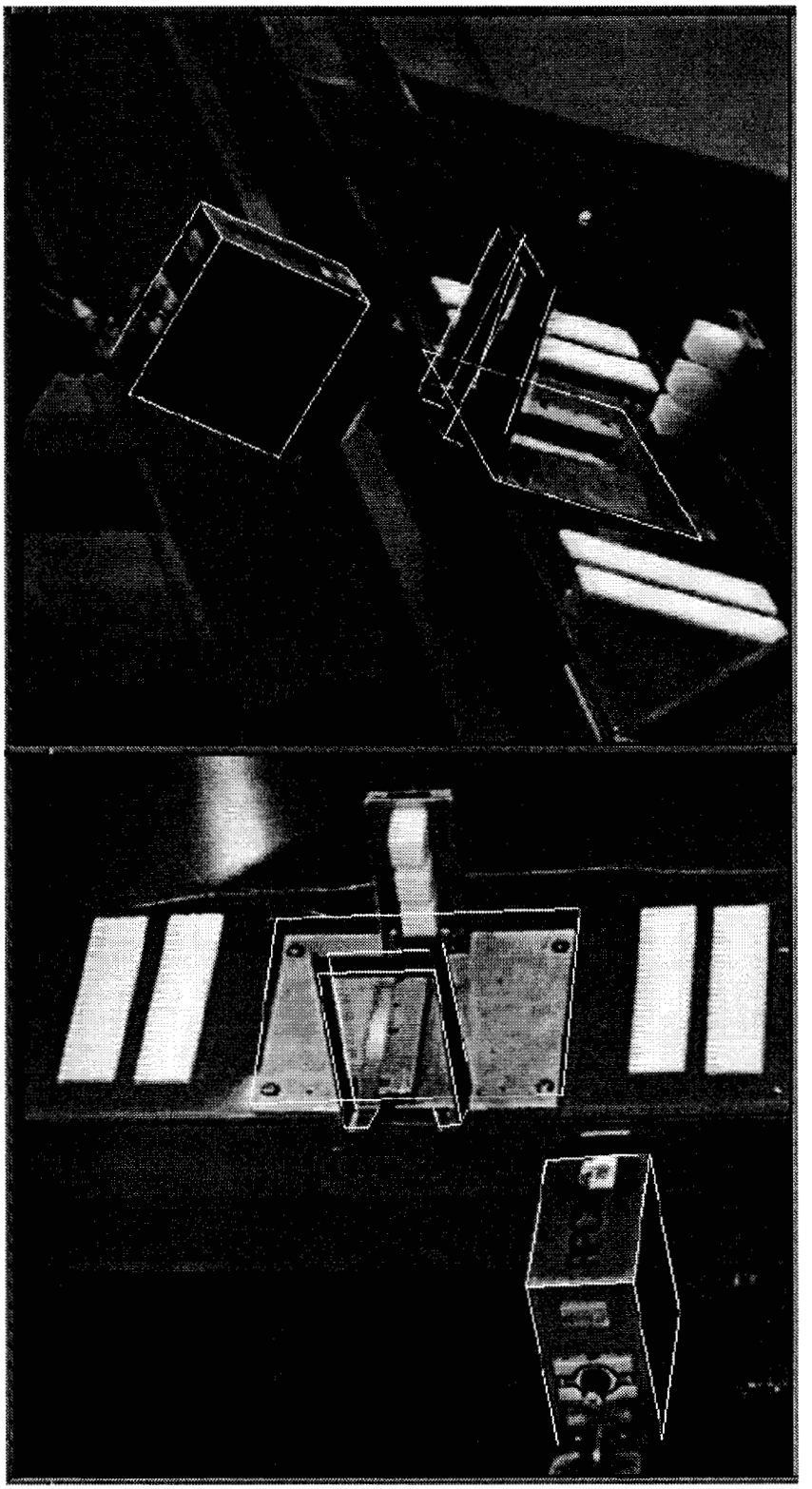

Figure 4: Receptacle localization using both camera views.

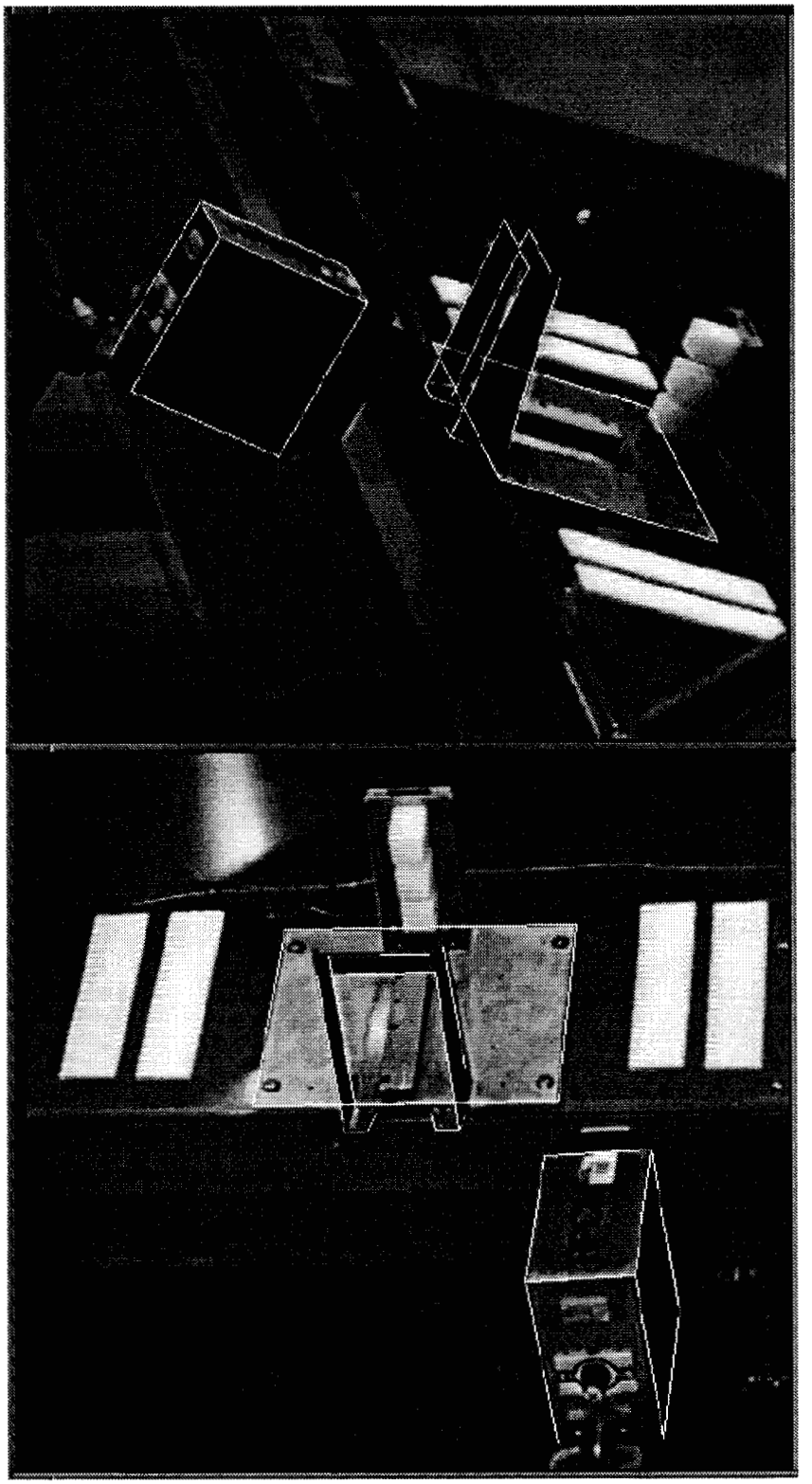

Figure 5: Simultaneous update using both cameras.

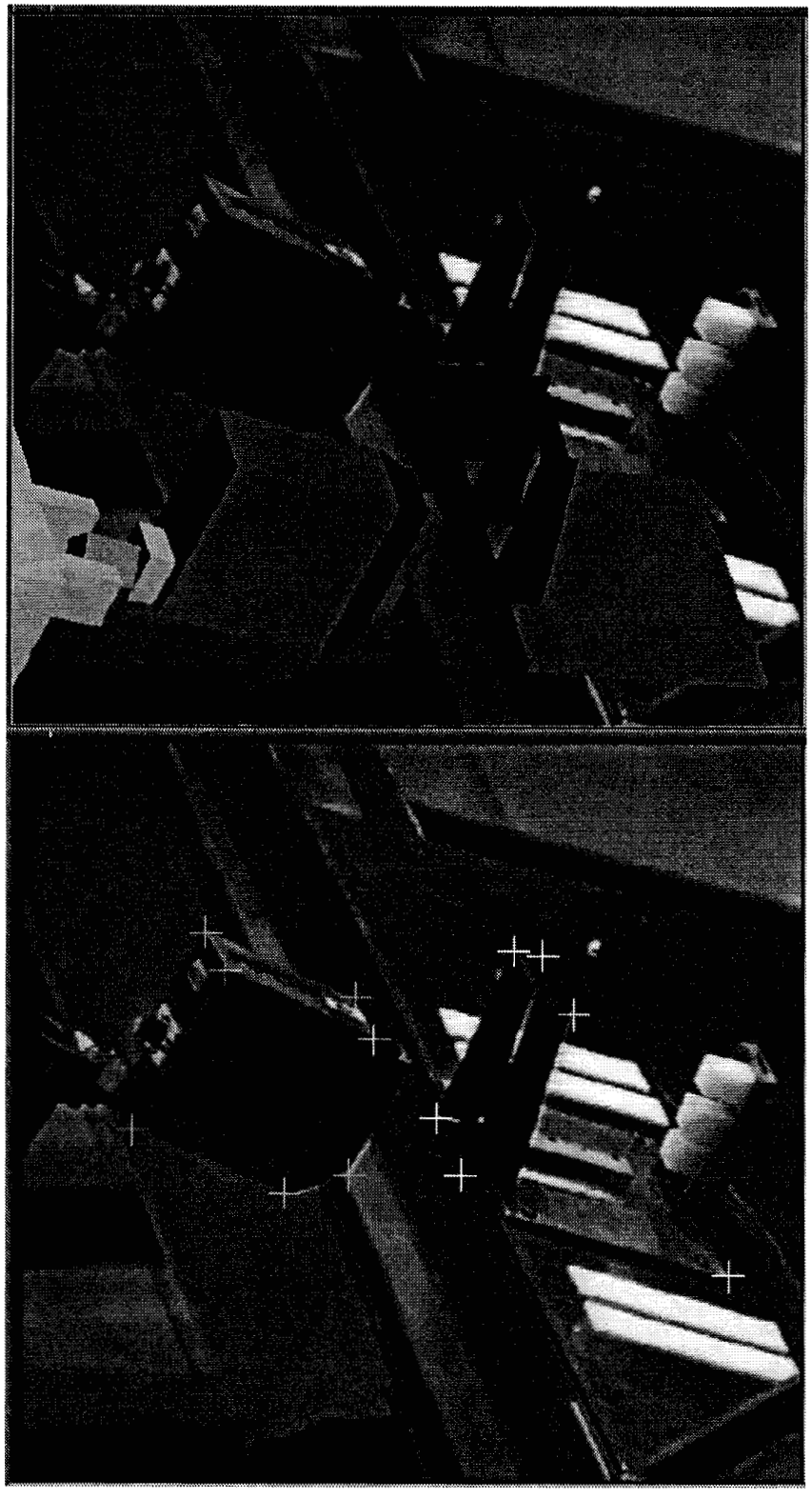

Figure 6: Corresponding points data entry for operator-interactive initial coarse matching.

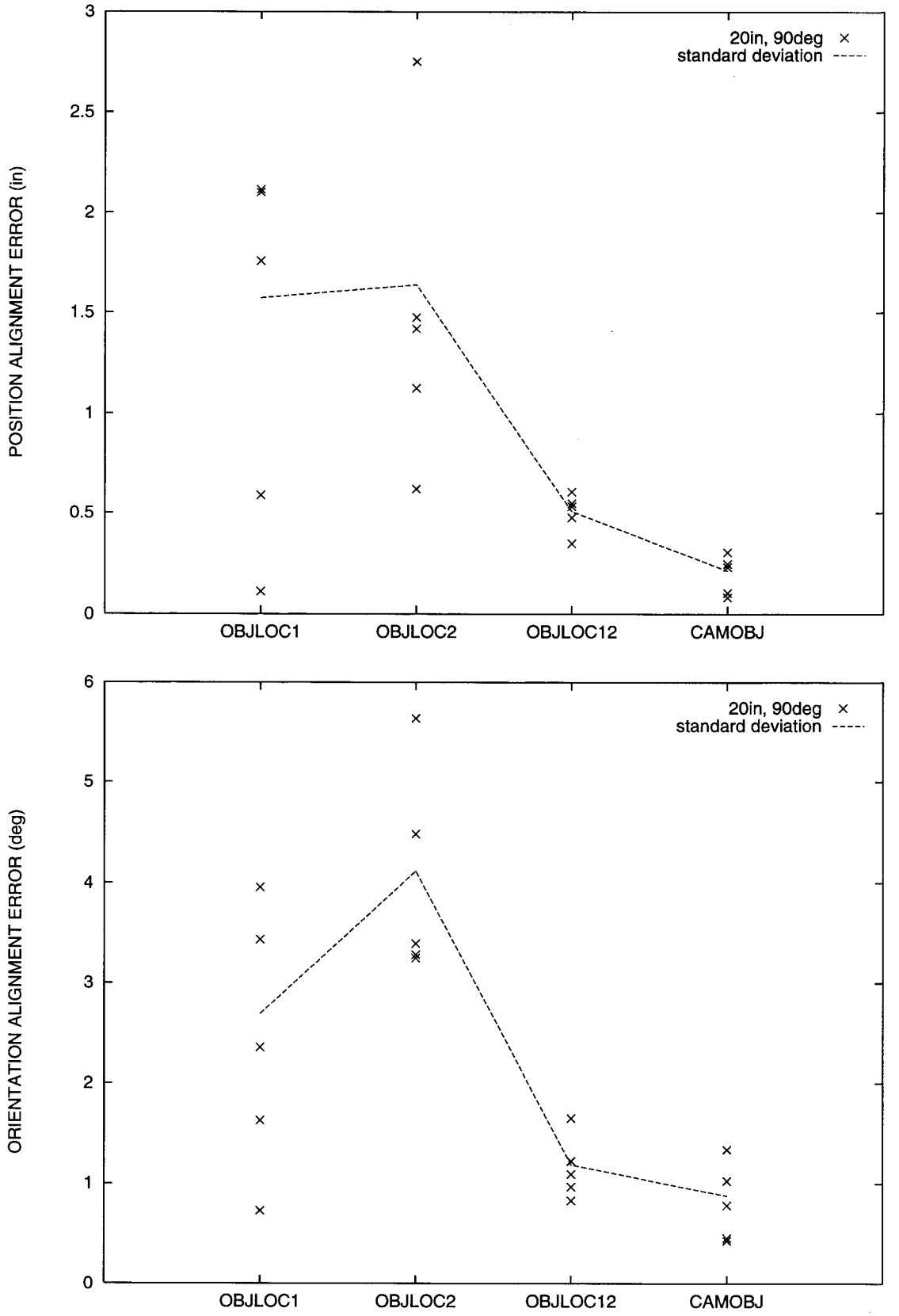

Figure 7: Alignment error plots comparing sequential and simultaneous updates

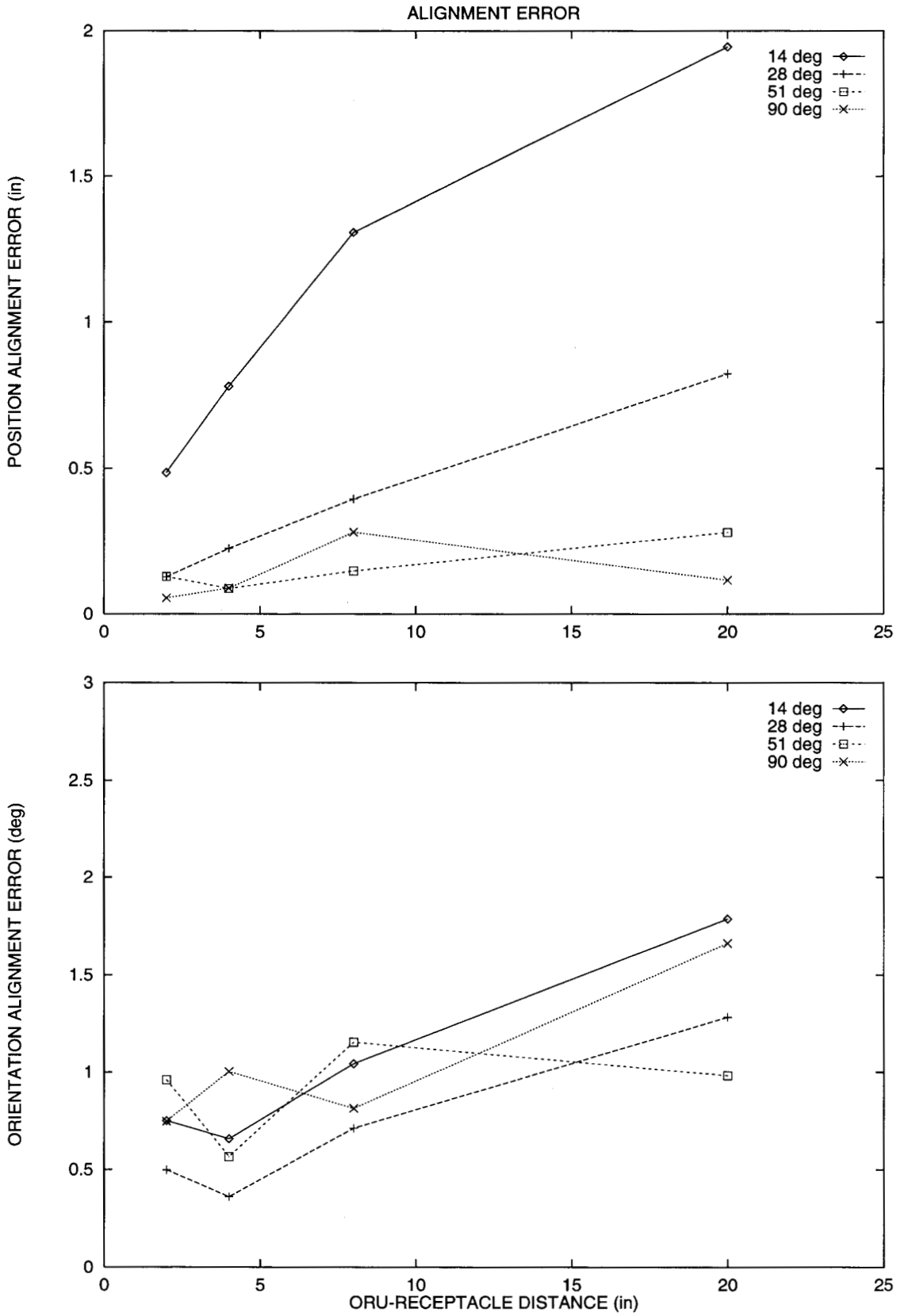

Figure 8: Alignment error with respect to the ORU distance from the receptacle

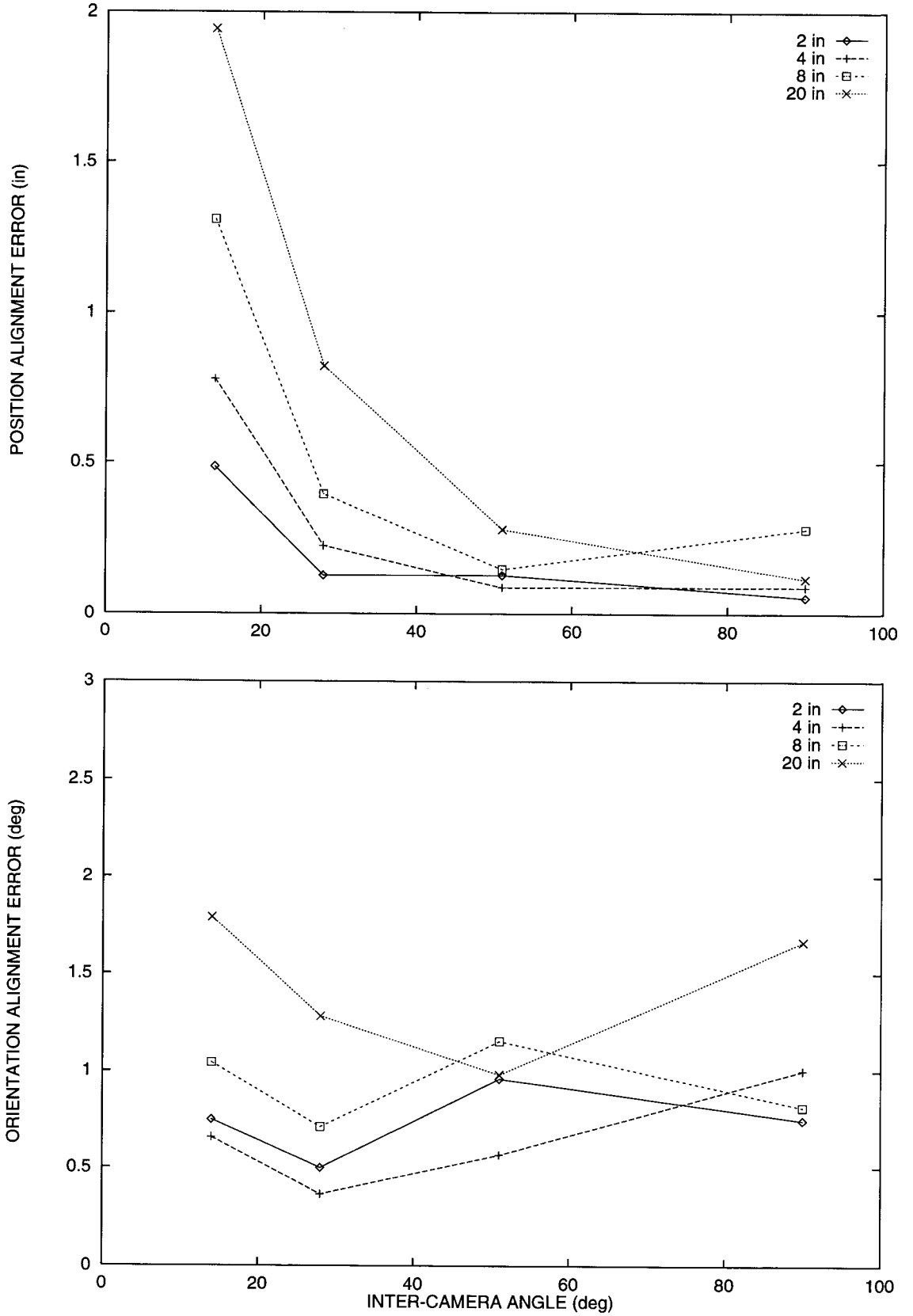

Figure 9: Alignment error with respect to the inter-camera angle

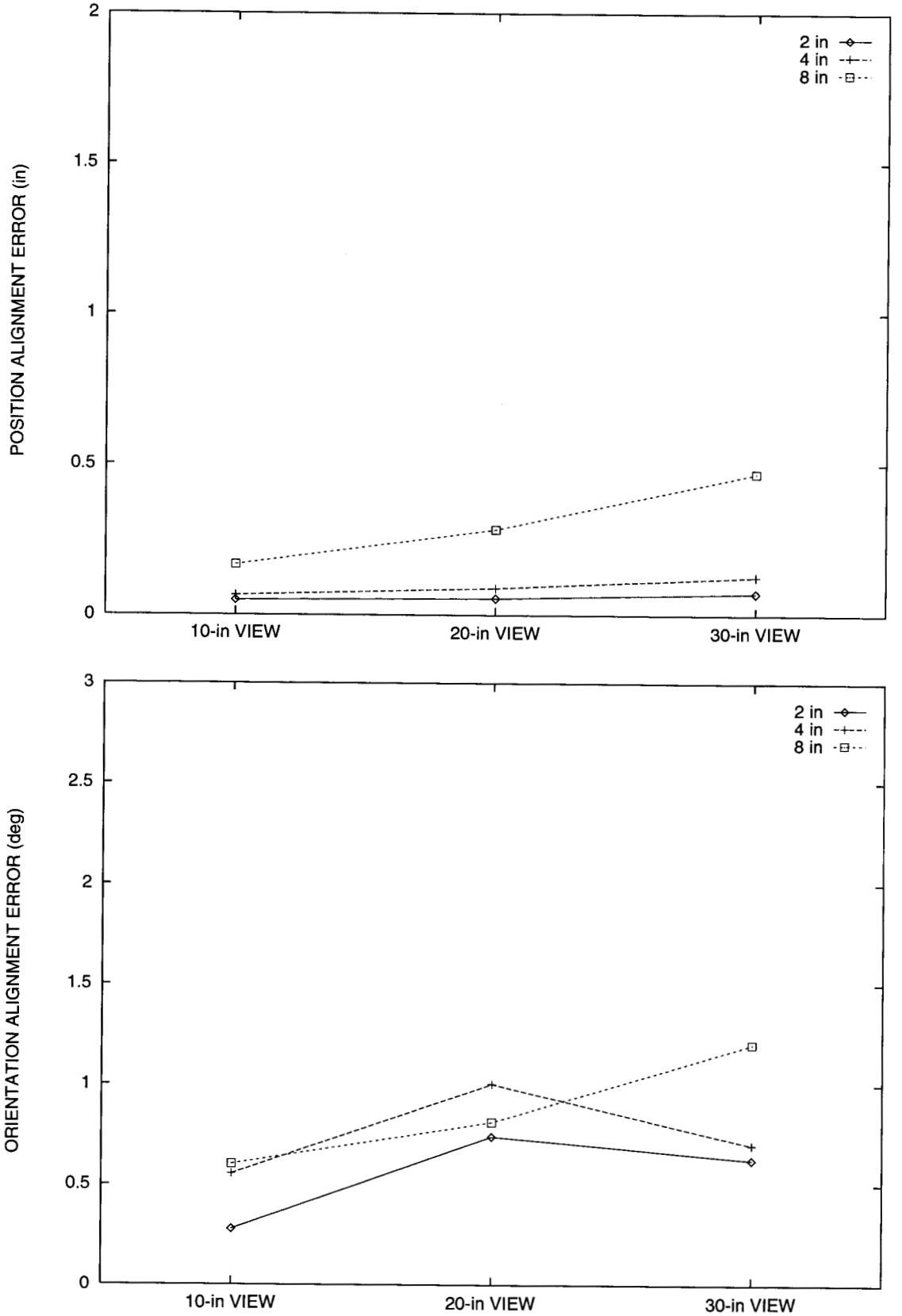

Figure 10: Alignment error with respect to camera view size

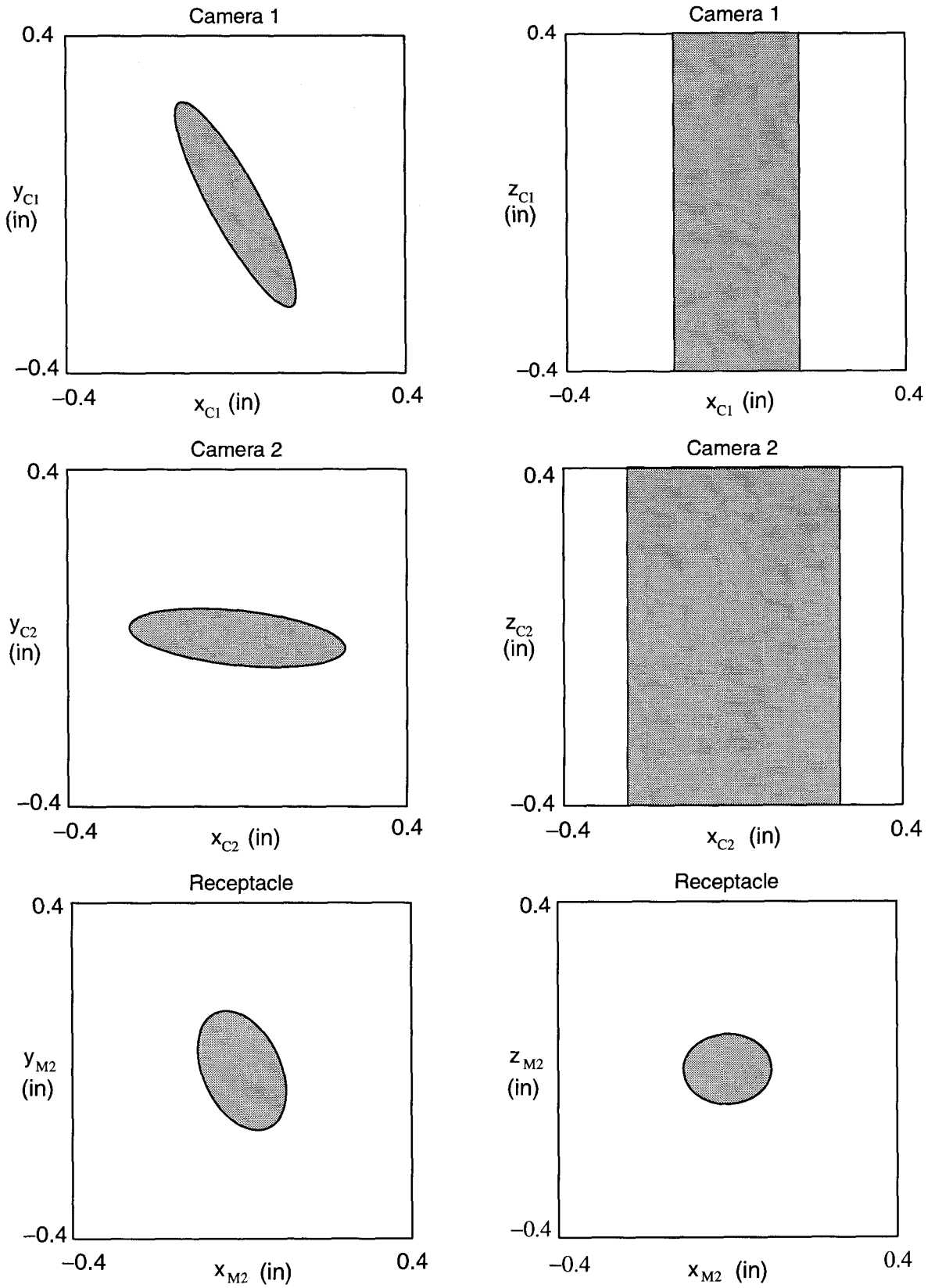

Figure 11: 1-σ ellipses of the pose estimates: (top) side camera pose relative to its camera frame, (middle) overhead camera pose relative to its camera frame, and (bottom) receptacle pose relative to the receptacle frame.